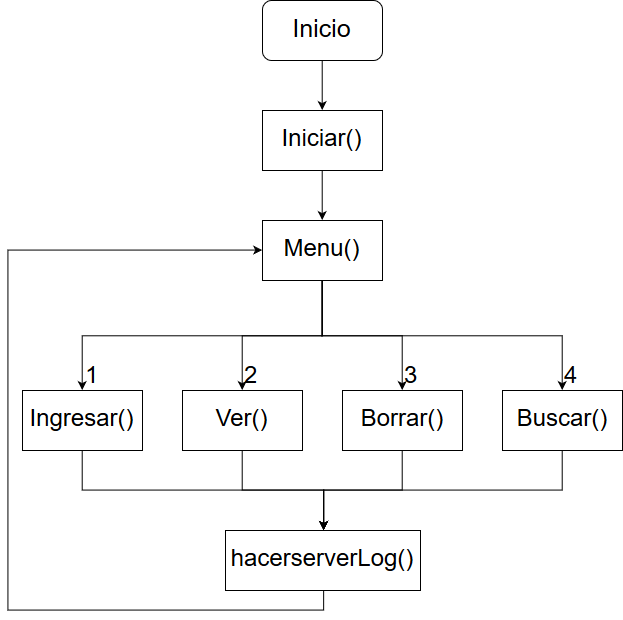
**Documentación Practica 2**

Alejandro Santamaría Pérez

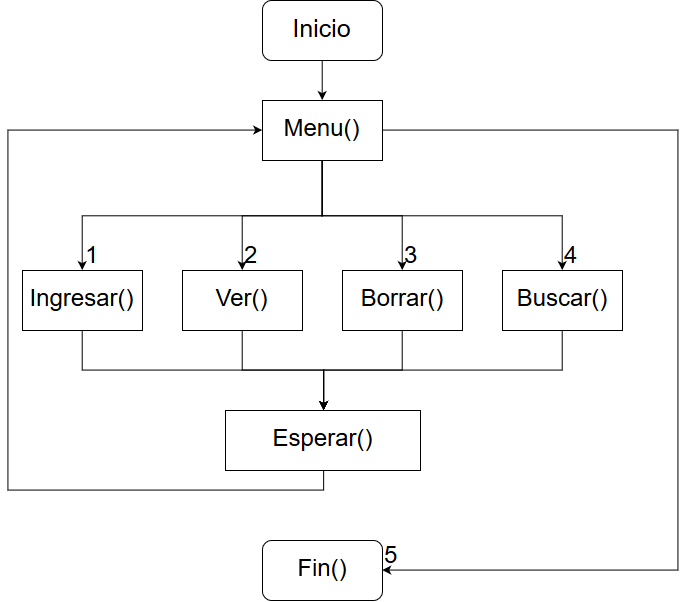
Juan David García Niño

**DIAGRAMA DE BLOQUES**

**Servidor:**



**Cliente:**



**DEFINICIÓN DE FUNCIONES**

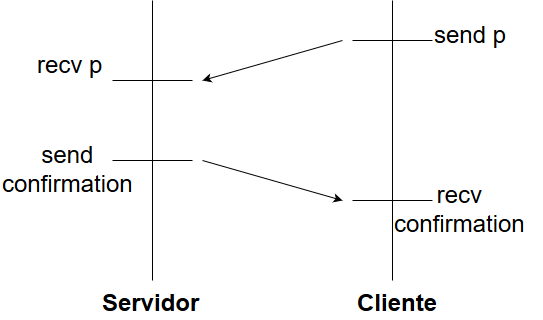
**INSERTAR REGISTRO:**

**Cliente:**

1. Se crea la estructura de tipo dogType y se asigna espacio en memoria.
2. Se reciben los datos digitados por el usuario para cada campo.
3. Se envían al servidor a través de sockets, utilizando la función send.
4. Se envía al servidor a través de send.
5. Se recibe mensaje de confirmación.
6. Se libera la memoria usada.
7. Se pide oprimir cualquier tecla para continuar.

**Servidor:**

1. Bloquea mediante mutex, por si cualquier otro hilo quiere acceder a los archivos que resultarán afectados.
2. Se crea una estructura del tipo dogType y se asigna espacio en memoria.
3. Se recibe la estructura (llena) de parte del cliente.
4. Se inserta la estructura al final del archivo dataDogs.dat.
5. Se libera la memoria usada
6. Se envía mensaje de confirmación
7. Se desbloquea mutex.



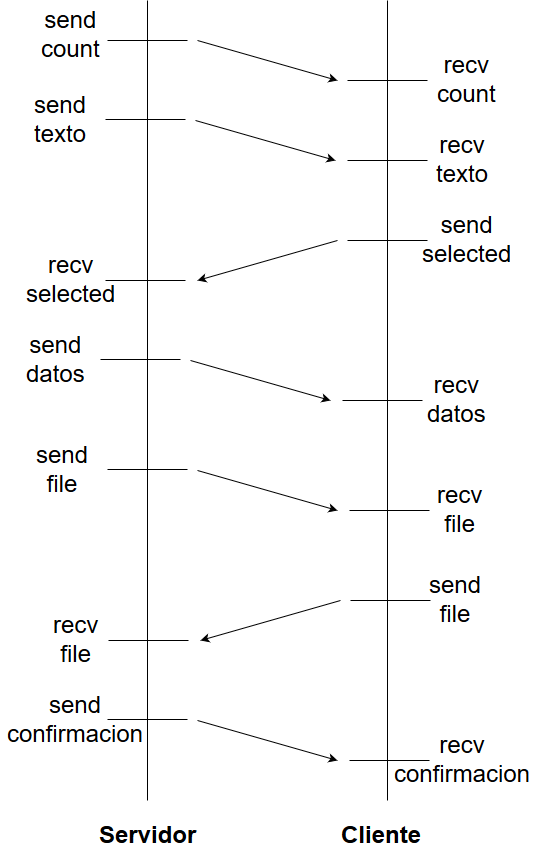
**VER REGISTRO:**

**Cliente:**

1. Recibe la cantidad de registros existentes.
2. Recibe e imprime mensaje de seleccionar registro.
3. Recibe por consola el índice que se desea ver.
4. Envía el índice al servidor.
5. Crea archivo temporal de la historia clínica de la mascota con datos recibidos del servidor.
6. Abre la historia clínica en el nano.
7. Envía historia clínica modificada al servidor.
8. Elimina archivo temporal.
9. Recibe mensaje de confirmación.

**Servidor:**

1. Bloquea mediante mutex, por si cualquier otro hilo quiere acceder a los archivos que resultarán afectados.
2. Cuenta los registros existentes en dataDogs.dat y envía dicho número al cliente.
3. Recibe el índice de la mascota que se desea ver.
4. Envía al cliente datos de la mascota seleccionada.
5. Envía archivo de historia clínica al cliente.
6. Elimina archivo de historia clínica.
7. Recibe archivo de historia clínica (modificado) del cliente, y lo guarda.
8. Envía mensaje de confirmación
9. Desbloquea mutex.



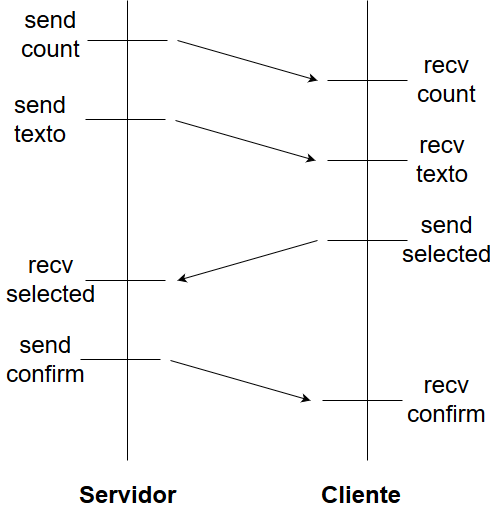
**BORRAR REGISTRO**

**Cliente:**

1. Recibe la cantidad de registros existentes.
2. Recibe textos.
3. Recibe por consola el índice que se desea borrar y lo envía.
4. Recibe mensaje de confirmación.

**Servidor:**

1. Bloquea mediante mutex, por si cualquier otro hilo quiere acceder a los archivos que resultarán afectados.
2. Cuenta la cantidad de registros existentes en dataDogs.dat, y envía el dato al cliente.
3. Envía texto al cliente.
4. Recibe el índice de la mascota que se desea eliminar.
5. Envía un comando al sistema para eliminar la mascota deseada.
6. Envía un mensaje de confirmación al cliente.
7. Desbloquea mutex.



**BUSCAR REGISTRO**

**Cliente:**

1. Recibe la cantidad de registros existentes.
2. Recibe texto.
3. Solicita el nombre de la mascota que se desea buscar y la envía al servidor.
4. Recibe un dato “flag” para saber si debe recibir un registro coincidente.
5. Recibe los registros de los perros con nombre coincidente.
6. Imprime los registros recibido.
7. Libera memoria.

**Servidor:**

1. Envía la cantidad de registros existentes en dataDogs.dat
2. Envía texto.
3. Recibe el nombre de la mascota que se desea buscar.
4. Revisa en el archivo buscando coincidencias.
5. Envía un dato “flag” al cliente.
6. De ser afirmativo el dato “flag”, envía el registro coincidente.
7. Libera memoria.

